(12) UK Patent Application (19) GB (11) 2 354 850 (13) A

(43) Date of A Publication 04.04.2001

(21) Application No 9922928.8

(22) Date of Filing 29.09.1999

(71) Applicant(s)
International Business Machines Corporation
(Incorporated in USA - New York)
Armonk, New York 10504, United States of America

(72) Inventor(s)
Roy B Harrison
Simon A Holdsworth
Thomas Quarendon
Michael George Taylor

(74) Agent and/or Address for Service
Kevin J Fournier
IBM United Kingdom Limited, Intellectual Property Department, Hursley Park, WINCHESTER, Hampshire, SO21 2JN, United Kingdom

(51) INT CL[7]
G06F 9/46 17/30

(52) UK CL (Edition S)
G4A AFGDC AFP

(56) Documents Cited
EP 0961452 A2 **US 5241305 A**

(58) Field of Search
UK CL (Edition R) **G4A AFGDC AFP**
INT CL[7] **G06F 9/46 17/30**
Online: WPI, EPODOC, PAJ, INSPEC, COMPUTER

(54) Abstract Title
Message broker using tree structures

(57) A message broker 12 provides for application integration amongst a plurality of data processing applications. The broker has a plurality of data processing nodes, where the data processing applications communicate by passing messages through data processing nodes of the broker. Messages received by the broker are transformed into a tree structure before being passed through data processing nodes of the broker. A list of destinations for a message is generated while a message passes through a data processing node of the broker. The list of destinations is also represented as a tree structure so that the list of destinations can be processed by the data processing nodes of the message broker.

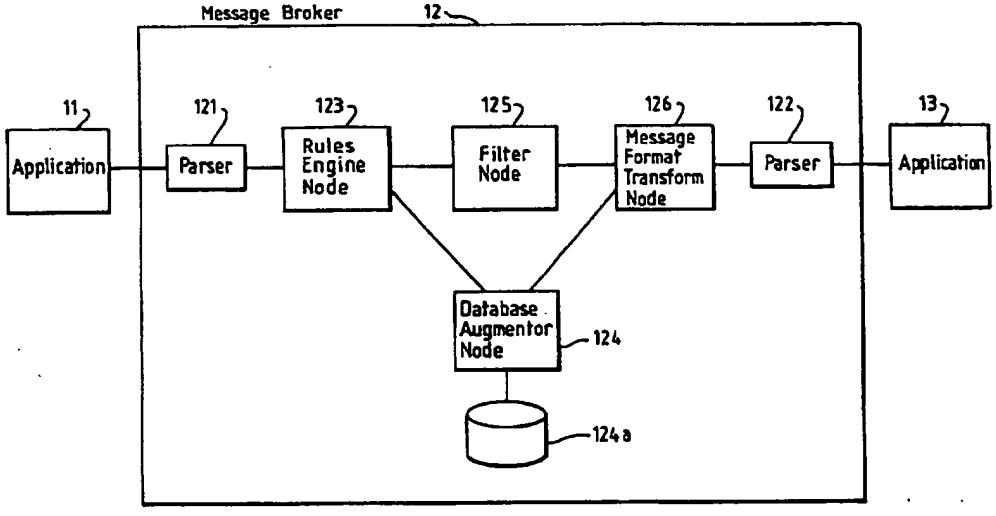

FIG. 1

GB 2 354 850 A

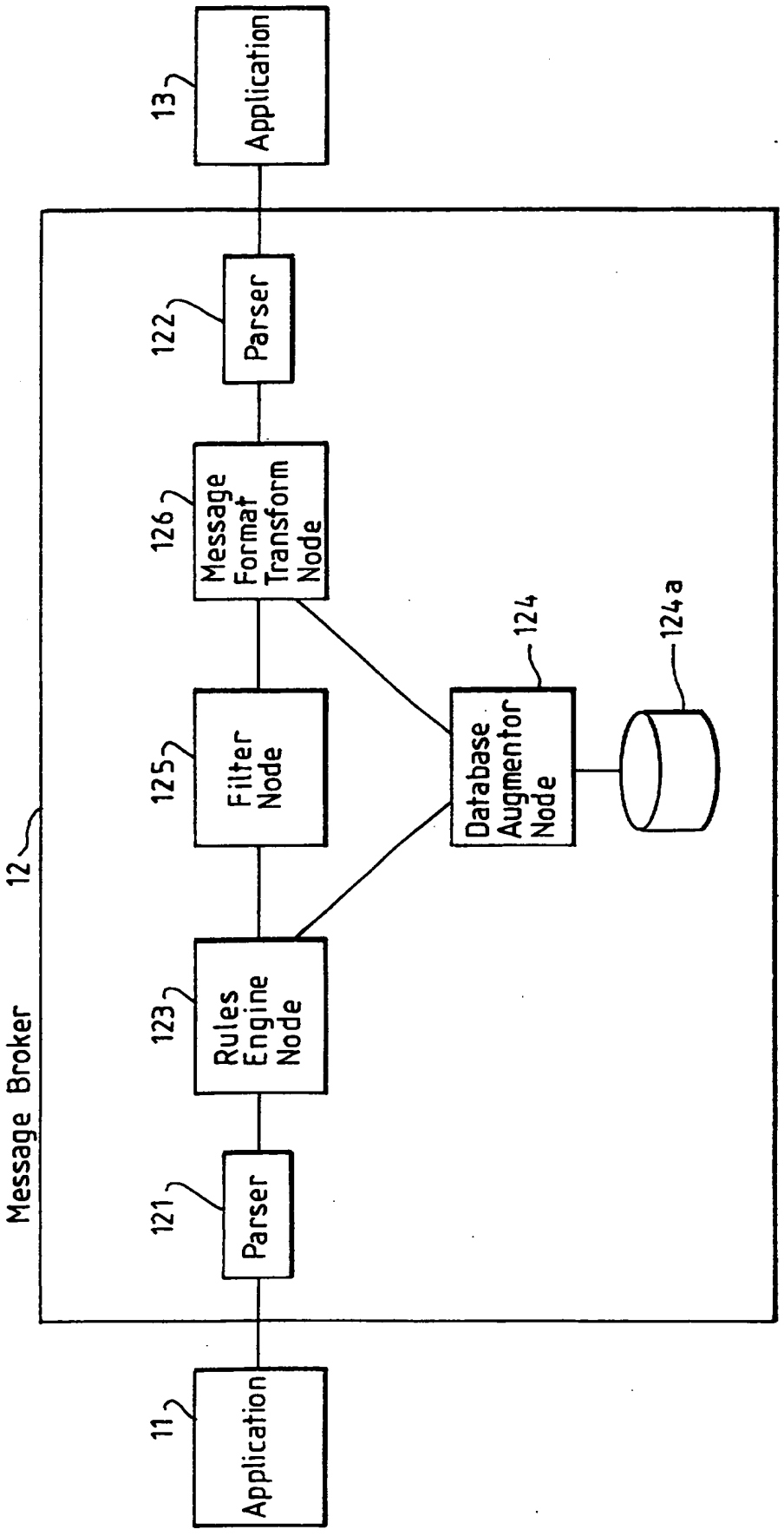
Application
13
Parser
122
Message Format Transform Node
126
Filter Node
125
Rules Engine Node
123
Parser
121
Application
11
Message Broker
12
Database Augmentor Node
124
124a

FIG. 1

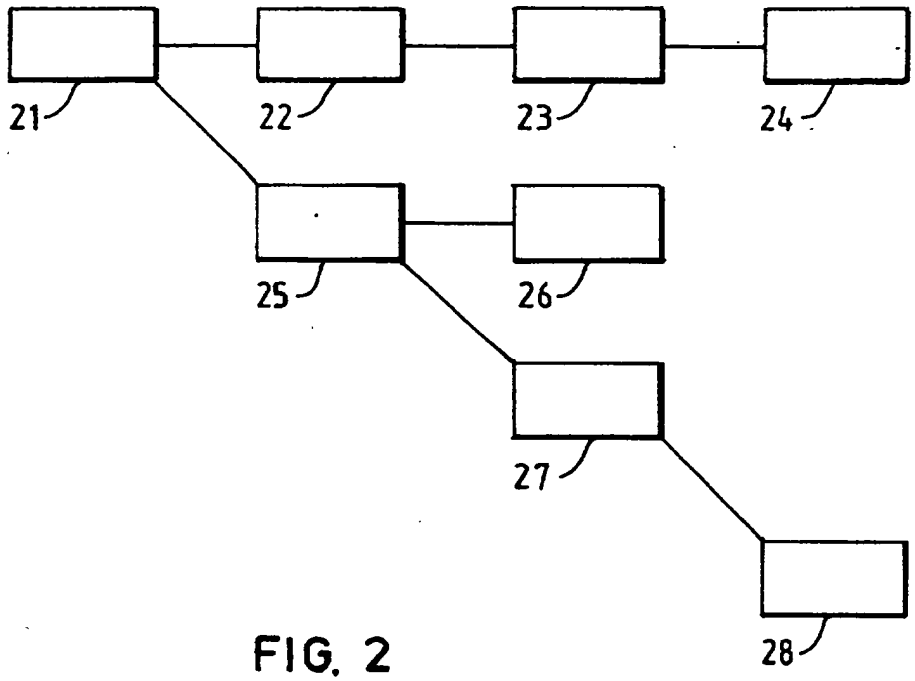

FIG. 2

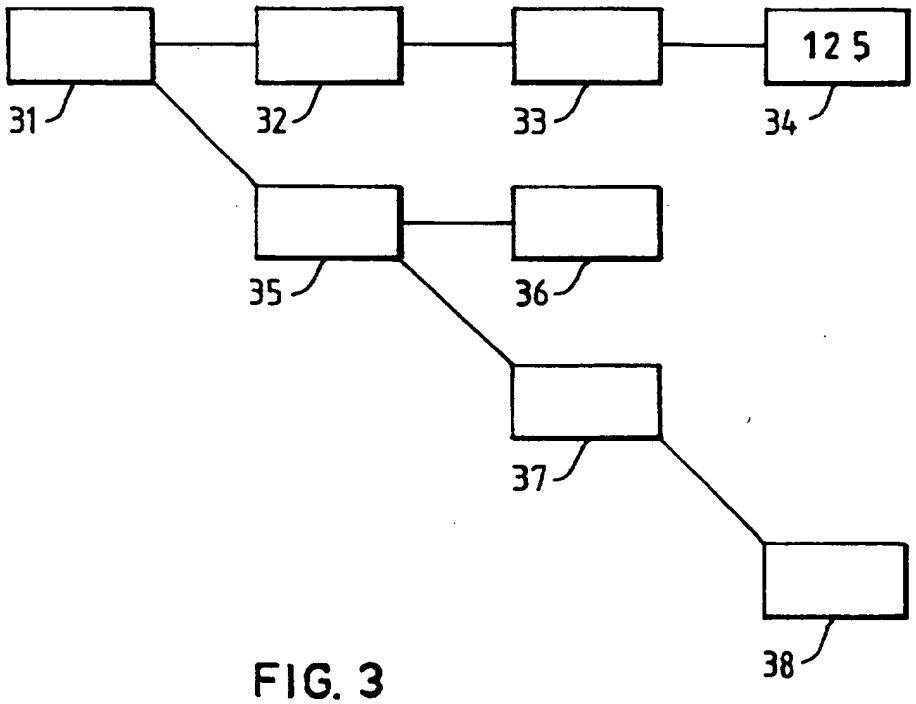

FIG. 3

DATA PROCESSING WITH REUSE OF EXISTING MESSAGE STRUCTURE TO ALLOW ACCESS TO DISTRIBUTION LIST

Field of the Invention

The present invention relates to the art of data processing, and more specifically to a message broker which receives messages from one data processing application, performs operations on such messages, and passes them along to at least an other data processing application.

Background of the Invention

Message brokers are becoming very common in the area of application integration. mMulti-function message broker systems are used to inter-connect applications which may be on heterogeneous platforms and may use different message formats. For example, Saga Software of Reston, Virginia (USA) (www.sagasoftware.com) have such a message broker product called "Sagavista" (a trademark of Saga Software). Further, Tibco Software Inc. of Palo Alto, California (USA) (www.tibco.com) also have such a message broker called "TIB/Message Broker" (both "TIB" and "TIB/Message Broker" are trademarks of Tibco). In these multi-function message brokers, a set of pluggable data processing nodes is provided, with each node being dedicated to a specific data processing task, such as message format transformation, publish/subscribe message distribution, and a rules engine for deciding (based on a plurality of predefined rules) where (e.g., to which further nodes in the message broker) an incoming message should be routed.

It is very common in such a message broker for a list of destinations for a message to be generated as a message passes through the data processing nodes of the broker. This list of destinations, sometimes called a distribution list, is generated by the broker as a result of predefined rules that are based on message content and other settings. The list of destinations tells the various parts of the broker where (e.g., which nodes) the message is to be forwarded to.

In the prior art, this distribution list was fixed such that once the list is generated it cannot be changed. This is problematic because it provides for a great deal of inflexibility. For example, th SMTP (Simple Mail Transfer Protocol) electronic mail, considers the distribution list as part of the message (i.e., the destination list is carried as a field(s) in the header of the message). The distribution list is then consulted by the data processing system to route the message

to its destinations. However, once generated, the distribution list cannot be altered, it is simply passed through the processing system in order to get the message to its destinations.

Thus, there is a great need in the art for more flexibility in dealing with distribution lists, once such lists are generated.

Summary of the Invention

According to a first aspect, the present invention provides a message broker for application integration amongst a plurality of data processing applications, the broker having a plurality of data processing nodes, where the data processing applications communicate by passing messages through data processing nodes of the broker. Messages received by the broker are transformed into a tree structure before being passed through data processing nodes of the broker. A list of destinations for a message is generated while a message passes through a data processing node of the broker. The list of destinations is also represented as a tree structure so that the list of destinations can be processed by the data processing nodes of the message broker.

Preferably, the list of destinations is provided separately from the message so that the list of destinations can be processed by data processing nodes separately from processing of the message. Also preferably, the messages received by the broker are transformed into a tree structure using format-specific parsers.

According to a second aspect, the invention provides a message broker for application integration amongst a plurality of data processing applications, the broker having: a plurality of data processing nodes, wherein the data processing applications communicate by passing messages through data processing nodes of the broker; a unit for receiving messages and transforming messages into a tree structure before passing the messages through the data processing nodes; and a unit for generating a list of destinations for a message while a message passes through a data processing node of the broker, wherein the list of destinations is also represented as a tree structure so that the list of destinations can be processed by the data processing nodes of the message broker.

According to a third aspect, the invention provides a message broker data processing method for application integration amongst a plurality of data processing applications, wherein the data processing applications communicate by passing messages through data processing nodes of the broker; the method comprising steps of: receiving messages from one application; transforming the received m ssages into a tree

structure before passing the messages through the data processing nodes; and generating a list of destinations for a message while a message passes through a data processing node of the broker, wherein the list of destinations is also represented as a tree structure so that the list of destinations can be processed by the data processing nodes of the message broker.

According to a fourth aspect, the invention provides a computer program product, stored on a computer readable storage medium for, when run on a computer, carrying out the method of the third aspect.

Thus, with the present invention, because the distribution list is represented in a tree format, modifications to the distribution list can easily be carried out using the existing data processing nodes of the broker. No modification to such nodes is required.

Brief Description of the Drawings

The present invention will be better understood through a description below of preferred embodiments thereof, which will be provided with reference to the following drawings:

Fig. 1 is a block diagram showing a message broker in which a preferred embodiment of the present invention can be advantageously used;

Fig. 2 shows a tree structure representation of a message used in a preferred embodiment of the present invention; and

Fig. 3 shows the distribution list represented as a tree structure, according to a preferred embodiment of the present invention.

Detailed Description of the Preferred Embodiments

In Fig. 1, application 11 is communicating with application 13 via message broker 12. Application 11, message broker 12 and application 13 are most likely running on separate machines and in communication with each other via a network (such as the Internet) but it is possible that all three applications are running on the same machine.

Application 11 sends XML (eXtensible Markup Language) format messages to message broker 12. When XML messages are received into message broker 12, a parser 121 parses the XML messages (which are in a flat byte structure needed by the transport mechanism used to link application 11 with broker 12) into a generic tree structure (a hierarchical structure, and thus no longer a flat structure). In fact,

no matter what format messages come into the message broker 12 they are parsed into this generic tree structure in the preferred embodiment. There is a corresponding parser 122 which parses messages from the generic tree structure into the COBOL message format (again in flat byte structure) required by application 13.

As shown in Fig. 2, the tree structure is made up of a sequence of data elements 21 to 28. These data elements may be simple data types, substructures, lists, arrays or simple data types or arrays of substructures. A substructure is itself composed of a sequence of named data elements of the same nature as the main message body, so the entire message can be expressed (and thus navigated) as a tree graph similar to a directory based file system but with an (optional) ordering of the entities within each directory. Each message has associated with it some meta-information describing its structure, content and physical representation. Note that the tree structure is an internal representation of the message's wire format. The wire format is converted to the tree by a parser. Every wire format can have a parser written to convert it into a tree structure. That tree structure might be degenerate, in that it only has one element, which contains the entire contents of the wire-format message, but the tree structure representation exists nevertheless.

Returning to Fig. 1, message broker 12 also includes a rules engine data processing node 123 which examines the content of messages passing through node 123 and applies various rules which have been pre-set by a systems administrator. As a result of a message having passed through the rules engine node 123, a distribution list is generated, which includes the identity of node(s) which the message should be next forwarded to upon leaving the node 123.

For example, one rule might be "if the message involves a stock price giving the current value of a company's shares on a stock exchange, then send the message to database augmentor node 124". Database augmentor node 124 receives the forwarded message and accesses a database 124a (in local storage) to determine how many shares the particular customer owns of the stock that is mentioned in the message and then this number of shares is added to the message, thus augmenting the original content of the message using the data from the database.

Another rule might be "if the message is marked confidential, then send message to filter node 125". Filter node 125 would then remove a field from the message which indicated the name of a party who originally sent the message so that when the message is received by a receiving application the name of the party sending the message is no longer ascertainable.

A common message broker would usually apply many such rules to messages and forward on messages to a large number of different choices of nodes in order to carry out intelligent message routing and processing. Only two rules have been shown in Fig. 1 for the sake of clarity.

Once the messages exit from nodes 124 and 125, they arrive at message format transform node 126 which carries out the function of transforming the format of the message from the XML physical representation format (used by application 11) to the COBOL physical representation format (of application 13). The basic structure of the message entering and exiting node 126 is still the hierarchical tree structure of Fig. 2, but the ordering of fields and data representations are transformed from that of XML to that of COBOL.

Upon exiting from node 126, the messages are then parsed from the generic tree structure to the COBOL structure (flat bytes needed by the transport mechanism used to link broker 12 with application 13, as opposed to a hierarchical tree structure). In the preferred embodiment, the transport mechanism used to link the applications 11 and 13 with the broker 12 is an asynchronous messaging and queuing transport mechanism, such as that of IBM's MQSeries (trademark) product.

When rules engine node 123 determines that a particular message is to be sent to filter node 125, an indication of the destination node 125 is represented as a tree-structure as shown in Fig. 3, which can then be passed to processing nodes of the broker in order to carry out any desired modifications of the distribution list. In this simple case the distribution list includes only one entry for destination node 125. Thus, data element 34, which is part of the tree structure of Fig. 3, includes the value 125. This then informs the broker that when the particular message leaves node 123 it should then be forwarded to each of the nodes on the distribution list (in this simple case, only node 125). The tree structure of Fig. 3 has been illustrated with many data elements because in the usual case there would be many destinations for a message, however, in the simple example provided, only one destination has been discussed.

Because the distribution list is represented as a tree structure, and messages passing through the broker also are represented as tree structures, modifications to the distribution list can easily be carried out using the existing nodes of the message broker. For example, the distribution list (represented as a tree structure) could be sent to a rules engine node (not shown in Fig. 1) and rules could be applied on the distribution list in order to generate another distribution list of destinations where the modified distribution list should be forwarded

upon leaving this rules engine node. Thus, the xisting data processing nodes of the broker can be easily used, without modification thereto, to perform data processing using the distribution list as input.

Preferably, the distribution list is not included in the message to which the distribution list pertains, but is instead provided in a separate tree structure (and thus the destination list is not sent along with the message to the destinations indicated in the destination list for that message). This allows the distribution list to be modified by nodes without requiring that such nodes be concerned with message data that is not distribution list data. For example, the rules engine node discussed in the preceding paragraph receives only distribution list data in a tree structure and thus the rules are applied only to the distribution list and not to fields of the message to which the distribution list pertains. This is advantageous because since the message has been parsed into a tree structure, any update to that tree structure has to be represented in the flat wire format of the message by walking the tree and rewriting the flat message. This is generally an expensive operation. Thus, preferably, the distribution list is not actually part of the message, and the distribution list is considered by the broker as a separate object.

CLAIMS

1. A message broker for application integration amongst a plurality of data processing applications, the broker comprising:

a plurality of data processing nodes, where the data processing applications communicate by passing messages through data processing nodes of the broker;

wherein messages received by the broker are transformed into a tree structure before being passed through data processing nodes of the broker;

wherein a list of destinations for a message is generated while a message passes through a data processing node of the broker, and the list of destinations is also represented as a tree structure so that the list of destinations can be processed by the data processing nodes of the message broker.

2. The message broker of claim 1 wherein the list of destinations is provided separately from the message so that the list of destinations can be processed by data processing nodes separately from processing of the message.

3. The message broker of claim 1 wherein the messages received by the broker are transformed into a tree structure using format-specific parsers.

4. A message broker for application integration amongst a plurality of data processing applications, the broker comprising:

a plurality of data processing nodes, wherein the data processing applications communicate by passing messages through data processing nodes of the broker;

means for receiving messages and transforming messages into a tree structure before passing the messages through the data processing nodes; and

means for generating a list of destinations for a message while a message passes through a data processing node of the broker, wherein the list of destinations is also represented as a tree structure so that the list of destinations can be processed by the data processing nodes of the message broker.

5. The message broker of claim 4 wherein the list of destinations is provided separately from the message so that the list of destinations can be processed by data processing nodes separately from processing of the message.

6. The message broker of claim 4 wherein the messages received by the broker are transformed into a tree structure using format-specific parsers.

7. A message broker data processing method for application integration amongst a plurality of data processing applications, wherein the data processing applications communicate by passing messages through data processing nodes of the broker; the method comprising steps of:

receiving messages from one application;

transforming the received messages into a tree structure before passing the messages through the data processing nodes; and

generating a list of destinations for a message while a message passes through a data processing node of the broker, wherein the list of destinations is also represented as a tree structure so that the list of destinations can be processed by the data processing nodes of the message broker.

8. The method of claim 7 wherein the list of destinations is provided separately from the message so that the list of destinations can be processed by data processing nodes separately from processing of the message.

9. The method of claim 7 wherein the messages received by the broker are transformed into a tree structure using format-specific parsers.

10. A computer program product, stored on a computer readable storage medium for, when run on a computer, carrying out the method of claim 7.

Application No: GB 9922928.8
Claims searched: 1-10

Examiner: Ben Micklewright
Date of search: 8 May 2000

Patents Act 1977
Search Report under Section 17

Databases searched:

UK Patent Office collections, including GB, EP, WO & US patent specifications, in:

UK Cl (Ed.R): G4A (AFGDC AFP)

Int Cl (Ed.7): G06F (9/46 17/30)

Other: Online: WPI, EPODOC, PAJ, INSPEC, COMPUTER

Documents considered to be relevant:

Category	Identity of document and relevant passage		Relevant to claims
A,E	EP0961452 A2	(IBM) See whole document	-
A	US5241305	(FASCENDA) See whole document, e.g. claim 1	-

X	Document indicating lack of novelty or inventive step	A	Document indicating technological background and/or state of the art.
Y	Document indicating lack of inventive step if combined with one or more other documents of same category.	P	Document published on or after the declared priority date but before the filing date of this invention.
&	Member of the same patent family	E	Patent document published on or after, but with priority date earlier than, the filing date of this application.